**CS 131C# - Beginner**

**HOP02 – Arrays and Enumeration**

12/29/2019 Developed by Kim Nguyen

10/10/2020 Revised by Kim Nguyen

Center for Information Assurance (CIAE) @City University of Seattle (CityU)



**Before You Start**

* Version numbers may not match with the most current version at the time of writing. If given the option to choose between stable release (long-term support) or most recent, please choose the stable release rather than beta-testing version.
* This tutorial targets Windows users and MacOS users.
* There might be subtle discrepancies along the steps. Please use your best judgement while going through this cookbook style tutorial to complete each step.
* For your working directory, use your course number. This tutorial may use a different course number as an example.
* The directory path shown in screenshots may be different from yours.
* If you are not sure what to do or confused with any steps:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

Students will be able to:

* Read and provide comments to code.
* Understand and use one dimensional and multidimensional arrays.
* Understand and use for loop.
* Understand and use Enumeration

**Resources**

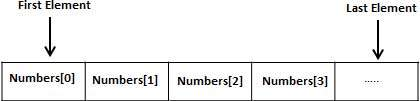
* C# Tutorials | W3Schools.com- <https://www.w3schools.com/cs/default.asp>
* C# Tutorials | tutorials.com- [https://www.tutorialspoint.com/csharp/](https://www.tutorialspoint.com/csharp/csharp_strings.htm)
* Multi-Dimensional Arrays in C Programming: Definition & Example | study.com - <https://study.com/>
* Loops in C# | geeksforgeeks.org - <https://www.geeksforgeeks.org/loops-in-c-sharp/>

**Part I: One Dimensional Arrays**

An array stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type stored at contiguous memory locations.

Instead of declaring individual variables, such as number0, number1, ..., and number99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables. A specific element in an array is accessed by an index.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

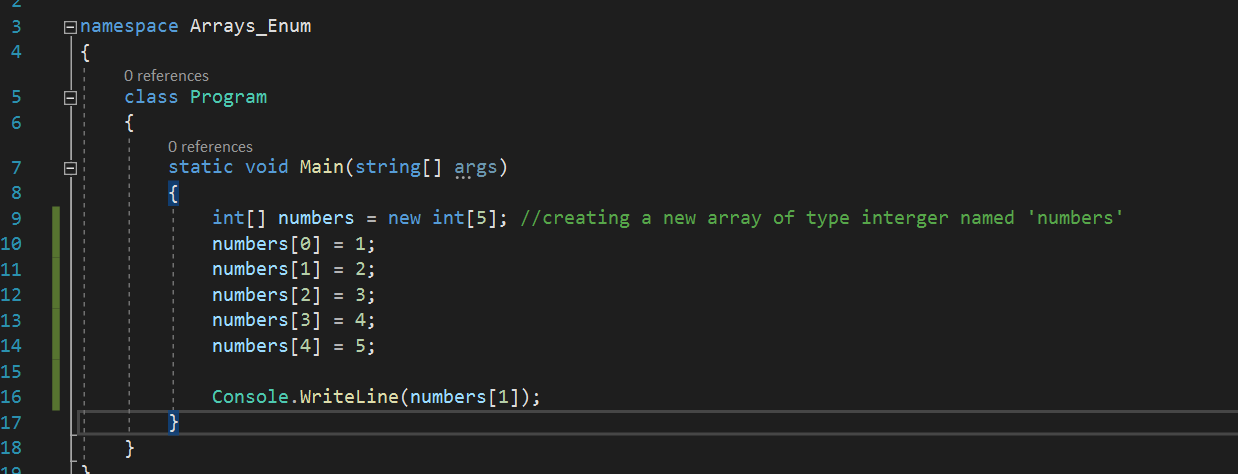


**Create a project**

1. Open Visual Studio.
2. File > New > Project
3. Select Console App (.NET Core), click Next
4. Type “Arrays” in the Project name and save it in the Module 2 of repository you cloned from week 1, it should be similar to below:

CS132/HOP-hands-on-practice-YourGitHubUserName/Module2

1) Type the following into Program.cs



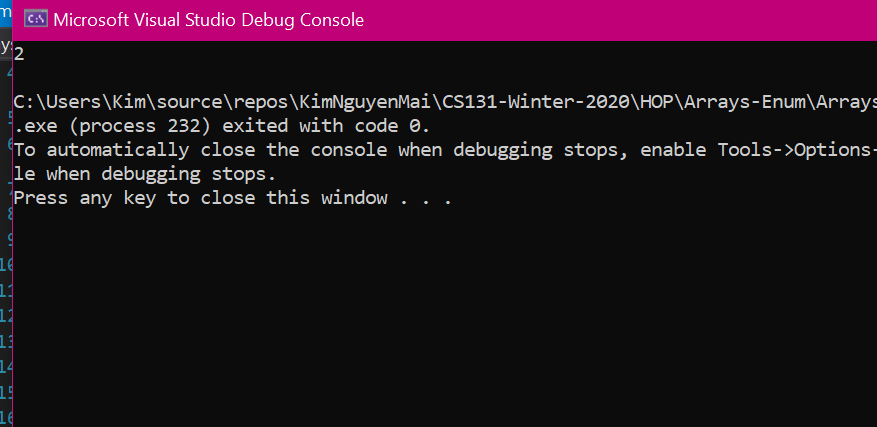
As you can see on line 9, there’s a green line of text, which is called a **comment** in programming:

Comments can be used to explain C# code, and to make it more readable. It can also be used to prevent execution when testing alternative code.

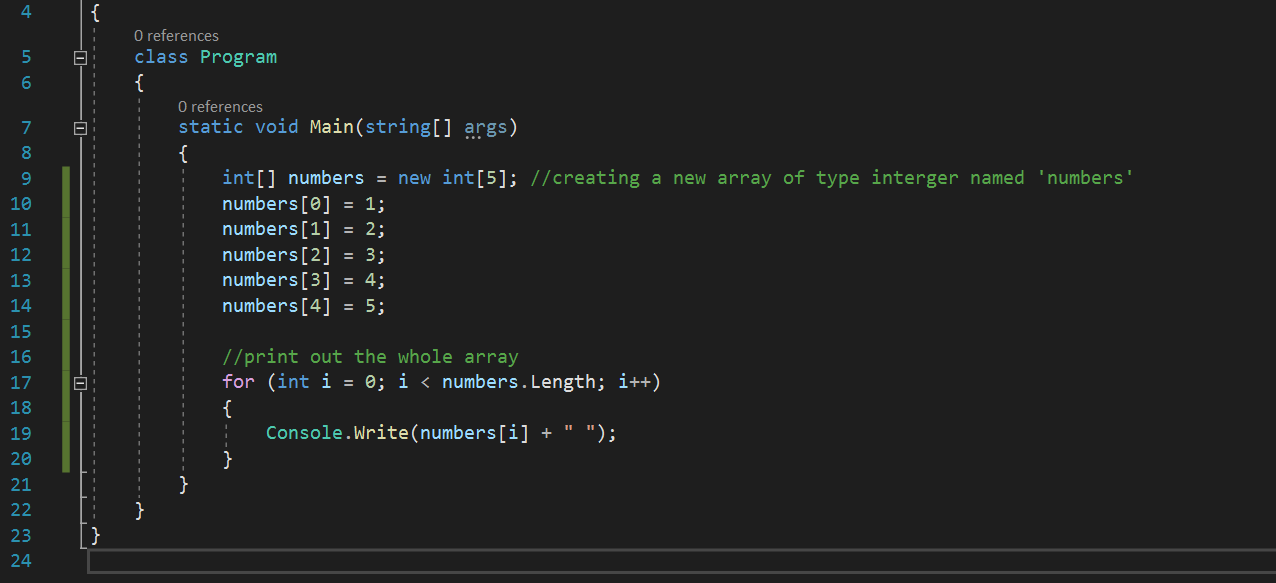
Single-line comments start with two forward slashes (//).

Any text between // and the end of the line is ignored by C# (will not be executed).

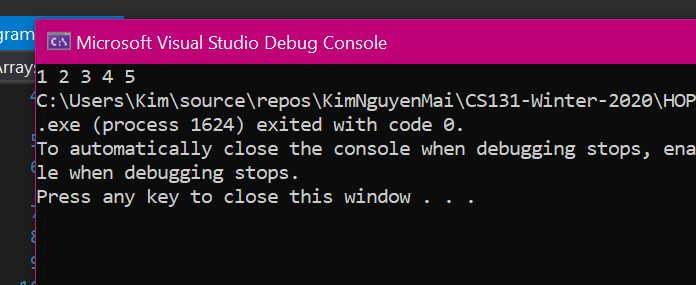
2) Run the program, you should see “2” on the console screen:



3) Modify your code to match the following screenshot:



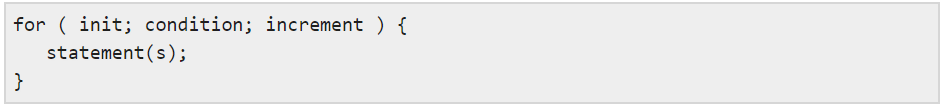
4) Run the program, you should see all elements of your arrays printed on the console screen:



**For Loop:**

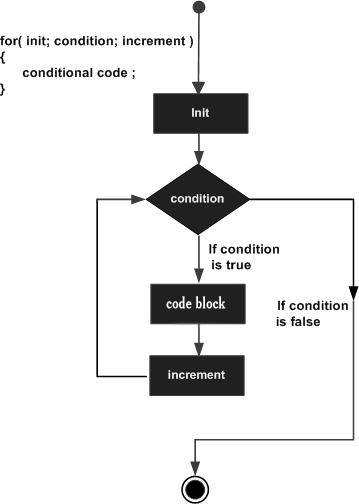
A **for** loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

The syntax of a **for** loop in C# is:

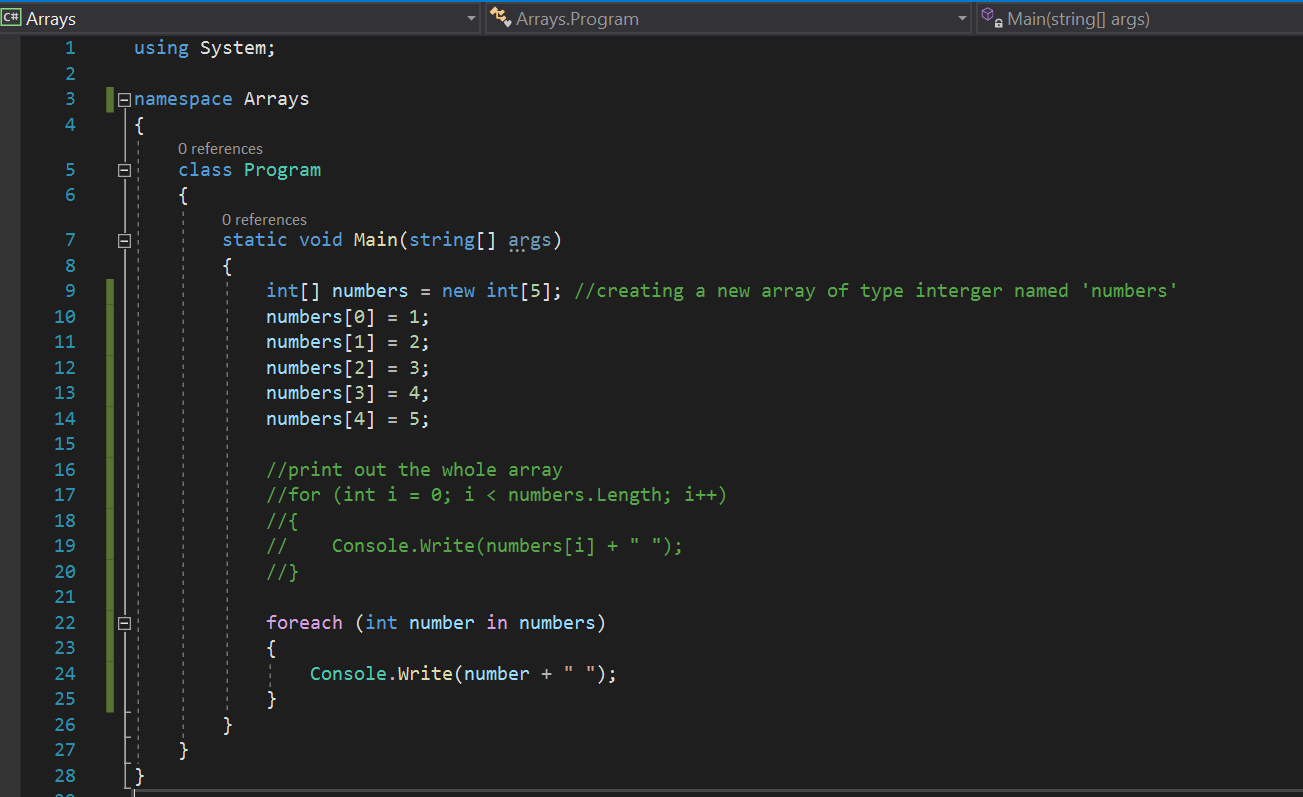


Here is the flow of control in a for loop −

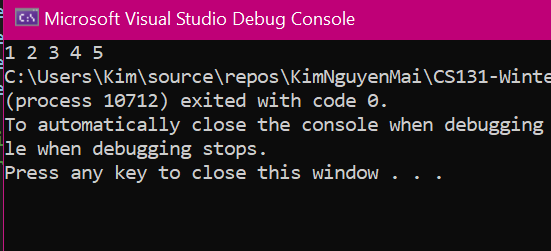
* The **init** step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
* Next, the **condition** is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and flow of control jumps to the next statement just after the for loop.
* After the body of the for loop executes, the flow of control jumps back up to the **increment** statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the condition.
* The condition is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then increment step, and then again testing for a condition). After the condition becomes false, the for loop terminates.



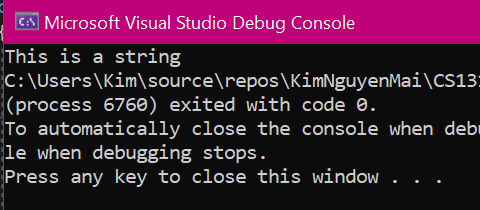
5) Let’s comment out the for loop by selecting the for loop and press Ctrl + K + C (to comment block of code). Then, modify your code to match the following screenshot:



6) Run the program, you should see all elements of your arrays printed on the console screen (this is just another way of printing the whole array):



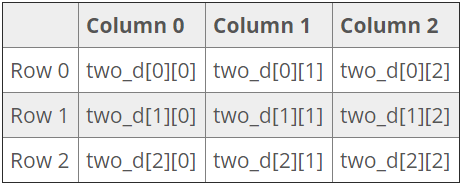
**Challenge**: **Create an array named “Sentence” of type string that has 4 elements: “This”, “is”, “a”, and “string” then print out the array using a foreach or for loop. Expected result:**



**Part II: Multidimensional Arrays**

A multi-dimensional array is an array that has more than one dimension. It is an array of arrays; an array that has multiple levels. The simplest multi-dimensional array is the 2D array, or two-dimensional array. It's technically an array of array

A valid type is required (in this case int), followed by the name of the array, and the size of each dimension. In this case, we created a 2D array that's 3 by 3 (three rows and three columns). As stored in the computer's memory, the array looks like this table.

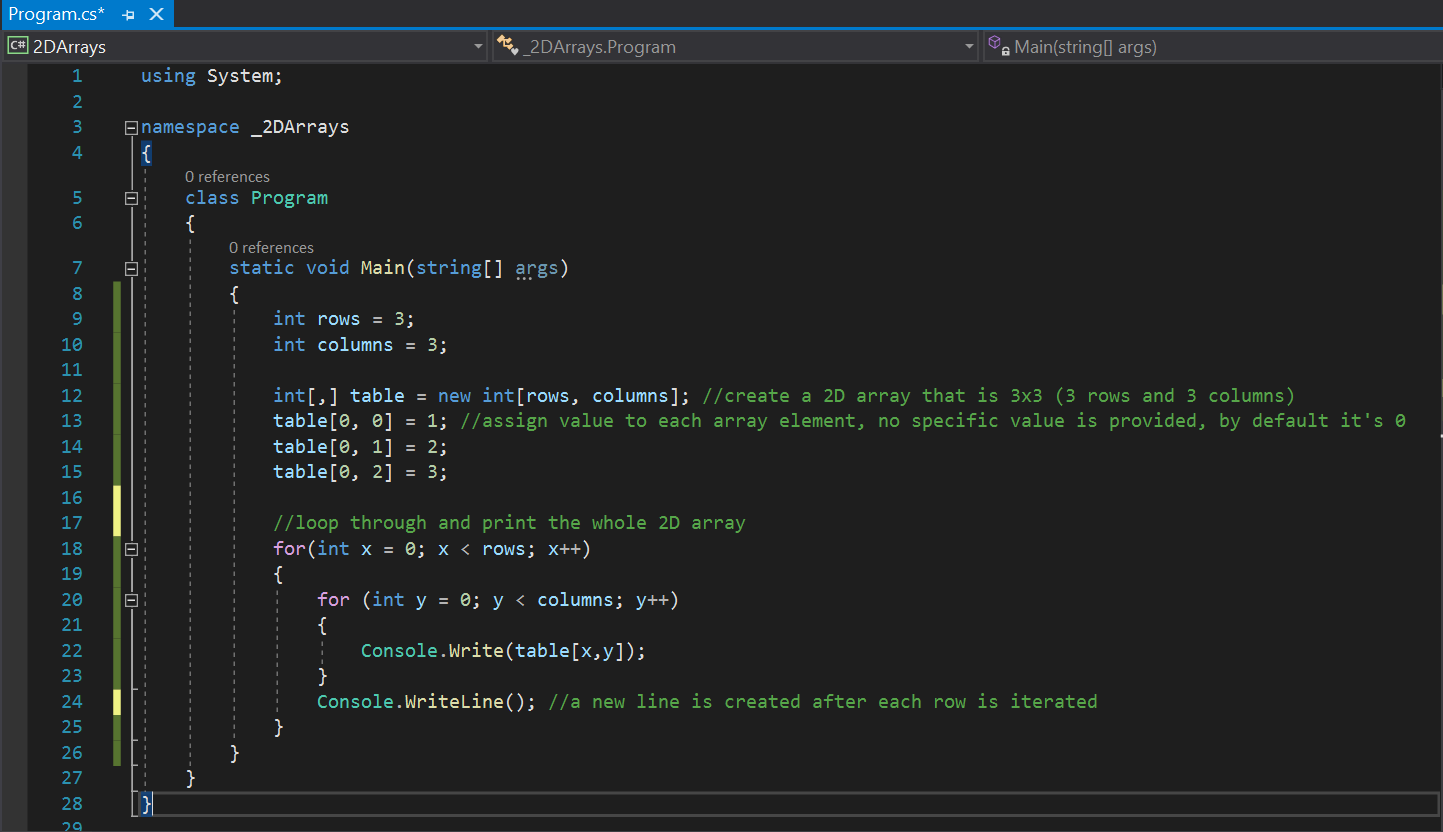


**Create a project**

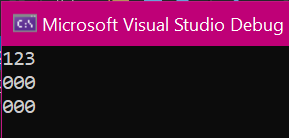
1. Open Visual Studio.
2. File > New > Project
3. Select Console App (.NET Core), click Next
4. Type “2DArrays” in the Project name and save it in the Module 2 of repository you cloned from week 1, it should be similar to below:

CS132/HOP-hands-on-practice-YourGitHubUserName/Module2

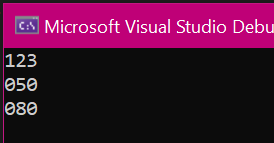
7) Type the following into Program.cs:



8) Run the program, you should see all elements of your arrays printed on the console screen in a table-like format:



**Challenge: Assign the value of 5 to the second element in the second row and 8 to the second element in the third (last row). Expected result:**

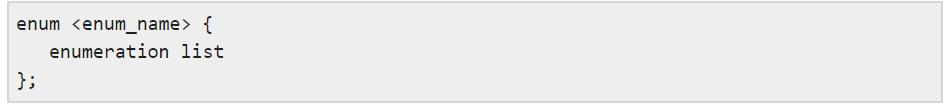


**Part III: Enumeration**

An enumeration is a set of named integer constants. An enumerated type is declared using the enum keyword.

C# enumerations are value data type. In other words, enumeration contains its own values and cannot inherit or cannot pass inheritance.

The general syntax for declaring an enumeration is



Where,

* The enum\_name specifies the enumeration type name.
* The enumeration list is a comma-separated list of identifiers.

Each of the symbols in the enumeration list stands for an integer value, one greater than the symbol that precedes it. By default, the value of the first enumeration symbol is 0. For example

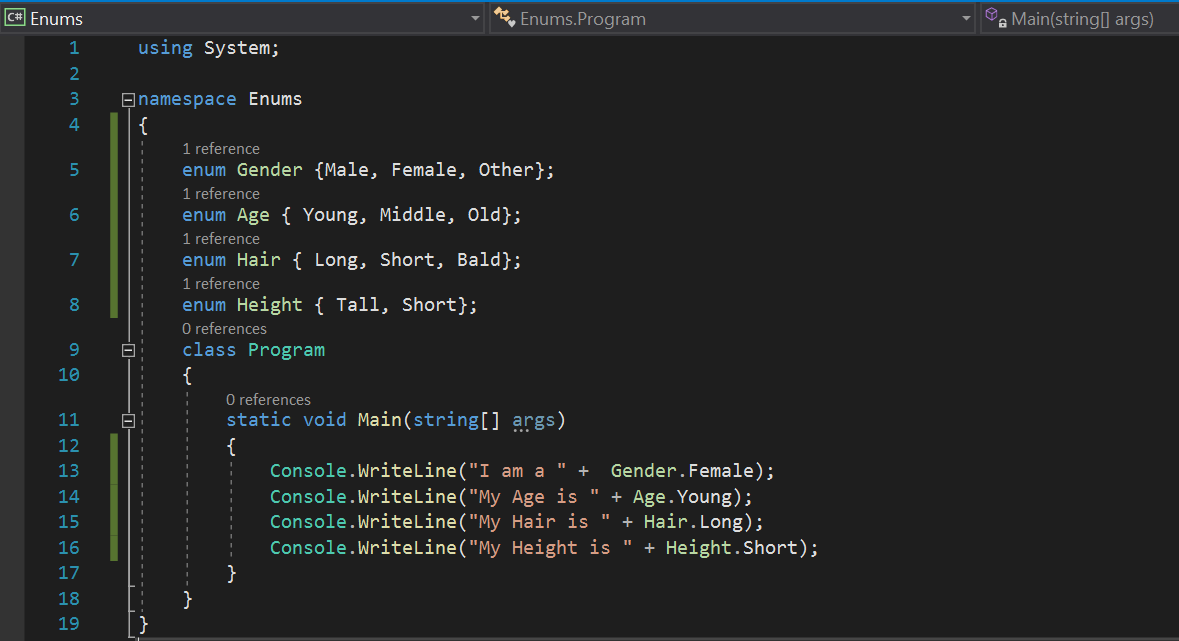
**enum Days { Sun, Mon, tue, Wed, thu, Fri, Sat };**

**Create a project**

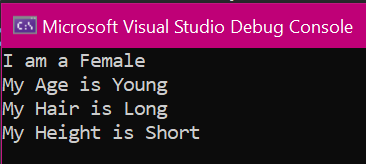
1. Open Visual Studio.
2. File > New > Project
3. Select Console App (.NET Core), click Next
4. Type “Enums” in the Project name save it in the Module 2 of repository you cloned from week 1, it should be similar to below:

CS132/HOP-hands-on-practice-YourGitHubUserName/Module2

9) Type the following into Program.cs:



10) Run your code, you should see:



No Challenge for this section 😊

**Push your work to GitHub**

Once you completed the Hands-on practice, do the following to push your work to GitHub

Go back to the Terminal (for Mac users) or Command Prompt (for Windows users), make sure you are in the right path, for example: KimNguyen/Desktop/CS132/CS132-HOP-Hands-On-Practice-KimNguyenMai/Module 2

Type the following command:

>>> git add . (to copy all changes you have made)

>>> git commit -m “Submission for Module 2 – YOUR GITHUB USERNAME” (To add a message to your submission)

>>> git push origin master (to upload your work to Github)